Constraints on long distance dependencies in gapping grammars

Patrick Saint Dizier

▶ **To cite this version:**

Patrick Saint Dizier. Constraints on long distance dependencies in gapping grammars. [Research Report] RR-0432, INRIA. 1985. inria-00076124

HAL Id: inria-00076124
https://inria.hal.science/inria-00076124

Submitted on 24 May 2006

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INRIA

CENTRE DE RENNES

IRISA

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P.105
78153 Le Chesnay Cedex
France
Tel. (3) 954 90 20

Rapports de Recherche

N° 432

CONSTRAINTS ON LONG DISTANCE DEPENDENCIES IN GAPPING GRAMMARS

Patrick SAINT-DIZIER

Juillet 1985

IRISA

INSTITUT DE RECHERCHE EN INFORMATIQUE
ET SYSTÈMES ALÉATOIRES

Campus Universitaire de Beaulieu
Avenue du Général Leclerc
35042 - RENNES CÉDEX
FRANCE
Tél. : (99) 36.20.00
Télex : UNIRISA 95 0473 F

Publication Interne n° 264

Juillet 1985

20 pages

CONSTRAINTS ON LONG DISTANCE DEPENDENCIES IN GAPPING GRAMMARS

Patrick SAINT-DIZIER

Research report JULY 1985.

ABSTRACT :

The goal of this paper is to show how Gapping Grammars, the most elaborated logic-based grammar formalism for processing natural language, can be restricted by the adjunction of constraints that limit the freedom of expansion of gaps and the freedom of establishing links between non-contiguous constituents.

RESUME :

Le but de ce document est de montrer comment les Gapping Grammars, le formalisme de grammaires logiques le plus elaboré à ce jour, peut etre restreint par l'adjonction de contraintes sur l'applicabilité des régles. Ces contraintes concernent la limitation de la liberté d'expansion des gaps et la liberté d'etablir des relations entre constituents non contigus.

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE
(L.A. 227)
UNIVERSITÉ DE RENNES 1 I.N.S.A. DE RENNES

INSTITUT NATIONAL DE RECHERCHE
EN INFORMATIQUE ET EN AUTOMATIQUE
(LABORATOIRE DE RENNES)

PI 252 **A model to analyse the causality in synchronous real time systems**
Albert Benveniste, 28 pages ; Avril 1985.

PI 253 **Précision numérique dans le cumul d'un grand nombre de termes**
Michèle Raphalen, Bernard Philippe, 44 pages ; Avril 1985.

PI 254 **Observational congruence of non-deterministic and communicating finite processes in asynchronous systems**
Boubakar Gamatié, 22 pages ; Avril 1985.

PI 255 **Dérivation d'algorithmes distribués d'arbitrage**
Jean-Pierre Verjus, René Thoraval, 30 pages ; Mai 1985.

PI 256 **Spécification et représentation par réseaux de Pétri d'un exécutif temps réel**
Maryline Silly, Houssine Chetto, 11 pages ; Mai 1985.

PI 257 **Towards an interactive Math Mode in TEX**
Jacques André, Yann Grundt, Vincent Quint, 16 pages ; Mai 1985.

PI 258 **Experiments in teaching METAFONT**
Jacques André, Richard Southall, 16 pages ; Mai 1985.

PI 259 **Une critique de la notion de test de processus fondée sur la non séparabilité de certaines classes de langages**
Philippe Darondeau, 40 pages ; Juin 1985.

PI 260 **Contrôler les transferts de connaissance dans les algorithmes distribués - Application à la détection de l'interblocage**
Jean – Michel Hélary, Aomar Maddi, Michel Raynal, 22 pages ; Juin 1985.

PI 261 **Détecter la perte de jetons et les regénérer sur une structure en anneau**
Michel Raynal, Gérardo Rubino, 24 pages ; Juillet 1985.

PI 262 **Elaboration et logiciel d'un indice de similarité entre objets d'un type quelconque. Application au problème de consensus en classification**
Israël-César Lerman, Philippe Peter, 72 pages ; Juillet 1985.

PI 263 **Mise en œuvre de l'analyse factorielle multiple pour des tableaux numériques, qualitatifs ou mixtes**
Brigitte Escofier, Jérôme Pagès, 58 pages ; Juillet 1985.

PI 264 **Constraints on long distance dependencies in gapping grammars**
Patrick Saint-Dizier, 20 pages ; Juillet 1985.

CONSTRAINTS ON LONG DISTANCE DEPENDENCIES IN GAPPING GRAMMARS

Patrick SAINT-DIZIER

Publication Interne

n° 264

Juillet 1985

The goal of this paper is to show, through several simple examples, how we can express and formalize some aspects of the natural language understanding process within the logic programming framework. More precisely, we will focus here on the more elaborated logic-based grammar formalism to date : the Gapping Grammars [Dahl and Abramson 84], [Dahl 84] and show how this formalism, which is largely too powerful to describe natural language sentences can be restricted by constraints and how these constraints are expressed. We explain how the freedom of expansion of gaps can be limited by introducing a case argument to gap variables. Next, we show how the freedom of establishing links between non-contiguous constituents (or movements of constituents, in transformational terms) can be limited by constraints. We take here the example of Ross' constraints.

For these specific problems, our approach is quite different of that of LFG [Kaplan 82] and GPSG [Gazdar 82]. These latter formalisms offer, in a first stage, few freedom to establish links between distant constituents, and then, they are augmented little by little by well defined more complex possibilities. For example, in LFG, c-domains were first defined. Later, it appears that they were too restrictive. Then, two new (and complementary) meta-variables were added : ==> and <==, to link two c-domains. These meta-variables allow one to describe long distance dependencies between two elements that belong to two different c-domains. With Gapping Grammars, the approach is completely different because the original formalism which allows any kind of link or movement is then restricted by appropriate constraints. For instance, at the level of long distance dependencies, any constituent can be linked to (or be in relation with) any other constituent. Then it is possible to block the application of a Gapping Grammar rule by appropriate constraints to avoid establishing prohibited links between constituents. This is precisely the goal of this paper.

In this paper, we first introduce very briefly Gapping Grammars. Section 2 is devoted to the introduction of a case argument in gaps. Finally, section 3 introduces the way long distance constraints are expressed in Gapping Grammar rules.

PAPIER RÉCUPÉRÉ ET RECYCLÉ

1_ GAPPING GRAMMARS :

1.1_ Definition :

Gapping Grammars (GG) are a generalization of the Metamorphosis Grammars (MG) [Colmerauer 78], [Dahl 77], Definite Clause Grammars (DCG) [F. Pereira 80] and Extraposition Grammars (XG) [F. Pereira 83]. A GG rule allows one to "indicate where intermediate, unspecified substrings can be skipped, left unanalysed during one part of the parse and possibly reordered by the rulés application for later analysis by other rules " [Dahl and Abramson 84], [Dahl 84]. The left hand part of a GG rule is composed of a non-terminal symbol followed by any string of non terminal and terminal symbols and gaps. PROLOG calls can also be used. The right hand side of a GG rule is a string of non terminal and terminal symbols, of gaps in any position and PROLOG calls. GG are a powerfull formalism that can be used to describe in a elegant and concise way complex natural language sentences as well as formal languages. A Gapping rule is of the form :

$$\alpha_1 \; gap(x_1)\, \alpha_2 \; gap(x_2) \; \text{---} \alpha_n \longrightarrow \beta$$

where: $\alpha_1 \in V_N$

$\alpha_i \; (i>1) \in V_N \cup V_T$

$x_i \in V_T^*$

$G = \{ gap(x_i), \; 1 \leq i \leq n-1 \}$

$\beta \in V_N^* \cup V_T^* \cup G^*$

For example, the left extraposition of an adjective in french is represented by the following GG rule :

```
Det Gap(X) Adj ---> Adj [,] Det Gap(X).
```

The basic form appears in the left hand part of the rule while the form with the movements appears in the right hand part. In fact, the arrow ---> has to be thought of as: <---.

Gapping Grammars are a notational tool to express movements of constituents or long distance dependencies. They are not transformational grammars. We can say, in fact, that the explicit constituents of a GG rule are in relation and that the type of relation involved is directly expressed by the GG rule. Gaps in Gapping Grammars has not to be confused with gaps or trace in linguistics. The term gap represents here varaibles that abbreviate symbols in grammar rules.

An efficient compiler of GG has been defined in [Popowich2 85] and unrestricted Gapping Grammars in [Popowich3 85]. In addition GG also appear to be well adapted to describe ID/LP grammars [Popowich 85]. Finally, some additional justifications of the formalism,

examples and comparisons with GPSG can be found in [Saint-Dizier 85].

1.2 How to write GG rules :

The formalism of GG is very powerful and allows one to describe in a single rule several phenomena which are, in fact, independent. Thus, it appears that a methodology for writing GG is strongly needed in order to have standard forms of rules and to assign a precise semantics to them. Writing "adequate " or "well-formed " GG rules is not the purpose of this paper, however, we give here two very simple restrictions which are useful to us in the remaining of this work.

(1) A GG rule describes a single relation between non-contiguous constituents (or a single transformation), even if this relation is complex, if it requires the presence of several explicit constituents which are rewritten into others and if several constituents or gaps are moved or deleted. In other words, by a single relation, we mean a relation that cannot be decomposed into several simpler ones, where each new relation can act independently of the others without any additional constraint of application. In addition, the restriction we state does not prohibit that a controller may have several controllee.

(2) Only the constituents that play a role in the phenomenon which is described in the GG rule have to appear in that rule. Conversely, all the constituents that play a role must appear in the rule.

In section 3, we will need another constraint on the way how to write GG rules. We need to make a distinction between gaps that represent any string that plays no role in the phenomenon described and gaps that replace an unknown string but which plays a role in the phenomenon, e.g. which is moved from its original location or deleted. We note this latter class of gaps : M-gaps (for Moving gaps). M-gaps are a convenient way to describe phenomena where some strings of words which are repositioned may be any string of words. On the other hand, gaps replace strings of words whose role is of no present interest. Thus, the GG rule writer will have to be able to make a distinction between 2 kinds of gaps. We think that this distinction is hardly desirable because the semantics of these two kinds of gaps is very different. In addition, the GG rule will also be more readable because of the greater precision of the notation.

As an example, consider the rule given in [Dahl and Abramson 84] to parse the sentence :
"The man with whose mother John left. "
this rule becomes :

NP(X) M-gap(Y) prep det gap(Z) prep(of) NP(X) --->
NP(X) prep [whose] gap(Z) M-gap(Y).

because "John left " is moved whereas "mother " is not and is of no present interest.

2 USING THE CASE ARGUMENT AS A RESTRICTION ON THE EXPANSION OF GAPS.

Traditionally, a case argument (subj, obj, mod, ...) in a grammar rule is used to specify the role the corresponding symbol plays in the structure described by the rule, and thus, the role the input string this symbol dominates plays in the sentence. This case argument intervenes in the construction of the output representation of the parser (for instance, to build a f-structure in LFG).

In GG rules, the case argument can also be used to constrain the application of a rule so as to prevent incorrect derivations. First, let's consider the adjunction of a case argument to explicit symbols. This adjunction is necessary, for instance, to allow the elision of the relative pronoun "who " in a sentence such as :

"The student I saw left. "
but not in :
** "The student saw me left. "*
where "who " is obligatory :
"The student who saw me left. "

This means that only relative pronouns that replace NP complements may be elided. The corresponding GG rules are :

*Rel-marqueur gap(X) NP(*case) --->*
*Rel-pronoun(*case) gap(X).*
Rel-pronoun(subj) ---> [who].
Rel-pronoun(obj) ---> [who] / [].
*(Symbols preceded by a * denote variables).*
(Terminal symbols are written between square brackets.)

It is also convenient to add the case argument to gap variables themselves. If we consider gap(X) as an element of the non-terminal vocabulary, we can then add arguments to it. From that point of view, a gap is seen as a variable which abbreviates a finite set of symbols. Adding a case argument to gaps implies that gaps represent any string of symbols that have a specific function, or that belong to a set of specific functions, if the case-argument may be a list.

The semantics of a case argument in a gap is that the nearest symbol, up in the tree, which dominates entirely the gap and to which a case function is assigned, has to have the function stated in the case argument of that gap.

We can express that a given gap has a specific function, i.e. the case argument is instantiated. We can also say that a gap has an unknown function but which is equal (or different) to that of another constituent or another gap. This can be done because the variables we use are logical variables [Palmer 83]. A gap is then noted :

gap(X(*case))

An illustration of this point is the extraposition of an adjective in a subject NP in french :

"Cette belle maison confortable est l'orgueil de ses proprietaires. "
(This nice, comfortable house is the proudness of its owners.)
which becomes :
"Comfortable, cette belle maison est l'orgueil des ses proprietaires. "

To prevent the extraposition of adjectives in object NPs, we can then write the following rule :

Det Gap(X(subj)) Adj(subj) --->
Adj(subj) [,] Det gap(X(subj)).

It is important to note that, even if we allow the case argument of a symbol to be a list (of functions it can or it cannot be), we do not describe ipso facto the complement of the language expressed by the rules. We only add an argument that plays the role of a restriction.

3_ LIMITATION OF THE MOVEMENT OF CONSTITUENTS.

3.1_ Introduction of constraints on the application of GG rules :

GG rules allow to define any relations between constituents in a sentence without any restrictions of application depending on the structure of the sentence. Thus, a lot of uncorrect sentences are recognized as well formed sentences. There exists constraints on establishing relations between constituents (or moving constituents). The most important constraints are Ross' [Ross 74]. These constraints have been developed within Chomsky's generative and transformational theory , but they remain valid for non-transformational theories with minor adaptations. The role of these constraints in GG is to block rules in certain derivational contexts.

We have chosen to consider here Ross' constraints because, as far as we know, they are the most general framework to express long distance dependency relation constraints. Recent extensions to

these constraints, we are aware of, can be expressed in the same formalism.

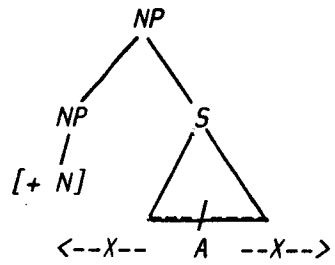
Our problem is not to discuss about the validity of Ross' constraints by themselves, but rather to build an adequate formalism for this type of contraints and to use Ross' constraints as an example. Our goal is then to show how they are expressed within the GG framework. The remaining of this paragraph is devoted to a brief presentation of Ross' constraints. Additional examples of constraints can be found in [Kaplan 82].

Ross' constraints applied on a GG rule require the knowledge :
(1) of the environment of constituents which are put into relation by this rule,
(2) of the ascendant nodes of the constituents we consider and their immediate daughters.

Examples :

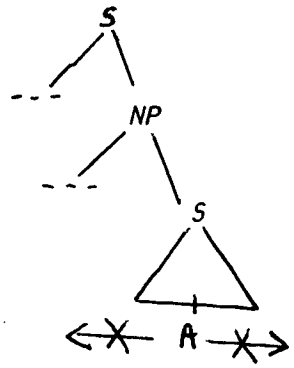
(a) The complex NP constraint :
"No element contained in a sentence dominated by a noun phrase with a lexical head noun may be moved out of that NP by transformation. "

To put it diagrammatically, this constraint prevents any constituent A from moving out of S :



(b) The sentential subject constraint :
"No element dominated by an S may be moved out of that S if that node S is dominated by an NP which is itself immediately dominated by S. "

This means that A cannot move out of the lowest S in the diagram :



3.2 Expression of long distance dependencies in GPSG and LFG :

Before we show how long distance dependencies and constraints on them are expressed in their generality in GG, we summarize in this short paragraph how they are taken into account in GPSG and LFG.

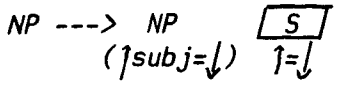
In GPSG, three kinds of rules intervene to establish links between non-contiguous constituents. Linking rules introduce gaps and holes, then derived categories are automatically produced and derived rules carry gaps and holes up in the tree. The last kind of rules are those that prevent the gap from percolating up past the node that immediately dominates the nearest derived category.

The use of this latter kind of rule to block a moved constituent from percolating up in the tree has the advantage of the simplicity, however, it seems to me that several complex cases cannot be taken into account. For instance, in any derivational context, no constituent that belongs to the domain of the node that immediately dominates a "dislocated " constituent may move out of this domain. It is possible to add mechanisms that take into account information about the derivational context, but this leads to severe modifications in the formalism and to very complex rules. In addition, derived categories rise several problems [Saint-Dizier 85]. Finally, it has never been made explicit how features of moved constituents interact with the rule-deriving machinery [Bear 83].

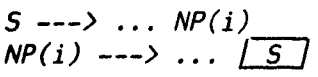
In LFG, long distance dependency relations (or constituent control , or syntactic binding), with respect to c-control, are expressed by bounding nodes. They are the root nodes of a domain and they are "boxed " in the Bresnan-Kaplan notation. A node that belongs to a given domain cannot move out of it. However, the effect of this boxing constraint can be circumvented by means of the linking equations which link two distinct c-domains [Kaplan 82].

In this case also, (a subset of) Ross' constraints are directly expressed in the rules. This means that, at a notational level, there is no longer a distinction between the description of the grammatical structures of a language and generalizations on that language such as these constraints. In addition, in a given grammar, the nodes that are roots of domains are defined a priori and once for ever.

Finally, and as far as we know, linking equations seems not to be adapted to express constraints that involve more than one ascendant. For example, the complex NP constraint can be taken into account by the rule :



whereas the sentential subject constraint cannot, because the lowest S has to be boxed only if its ascendants are NP and S. This set of ascendants cannot be expressed in a single rule. A way to take this into account in LFG is to create "indexed " categories that will take into account the ascendant nodes of a symbol, directly at the level of the grammar. If we add an arbitrary number to symbols to create new indexed symbols, then , the sentential subject constraint is expressed as follows :



and S --->... NP remains valid for other contexts.

But, by creating indexed categories, we heritate of some of the problems of GPSG. Nevertheless, LFG can take into account several long distance dependency contraints with a formalism which is efficient and easy to use.

3.3 Expression of Ross' contraints in Gapping Grammars :

In the GG rule context, the expression of Ross' constraints is divided into five sub-problems :

(1) The way how to memorize the derivational context of each symbol of a rule. The derivational context is the set of rules (or derivations) that have been applied from the starting symbol of the grammar to reach the symbol we consider for the sentence being parsed. We call these rules the ascendant rules of the symbol. The derivational context of each symbol of a rule is used here to know if the GG rule we consider may be applied (i.e. it is not equal to the blocking context of a constraint) or if it may not.

(2) The definition of the domain, for a given constraint, in which any constituent of this domain may move without any restriction. We call this domain the R-domain of the constraint.

(3) The way how to represent a constraint. To each constraint is associated a list of rules (or derivations) we call the blocking context of the constraint. These rules are the ordered list of derivations described in the domination conditions. We also introduce the notion of a constraint R-consistent for a given GG rule.

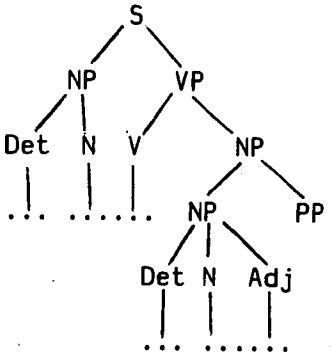
(4) The detection that a constituent (an explicit one or a M-gap) of a given R-domain moves out of this domain or is in relation with a constituent out of that domain. We say that, in

this case, the constraint is R-valid for the GG rule we consider and for the sentence being parsed.

(5) The expression of a constraint in a GG rule, with two points of view for its implementation : static and dynamic.

3.3.1_ Memorizing the derivational context of a symbol :

The **derivational context** of a symbol in a given rule and for a precise sentence is the ordered list of the rules that have been applied to reach it from the starting symbol of the grammar. For example, if we have the following tree :



then, the derivational context of the symbol Adj are the rules :

S --> NP VP
VP --> V NP
NP --> NP PP
NP --> Det N Adj
That we represent, for more convenience, by the following list of pairs :

(S,NP.VP).(VP,V.NP).(NP,NP.PP).(NP,Det.N.Adj)

The derivational context is built up in the GG rules in a specific argument, added to each explicit symbol and M-gap, and noted : *dc. The only rules that are taken into account to build it are those who operate a reduction. This means that rules that only reposition symbols are irrelevant for the derivational context. For example, the rules :
S ---> NP VP.
NP ---> Det Adj N.

becomes :
S(*dc) ---> NP(*dcl) VP(*dcl) Concatenate(*dc,(S,NP.VP),*dcl).

*NP(*dc) ---> Det(*dcl) Adj(*dcl) N(*dcl)*
*Concatenate(*dc,(NP,Det.Adj.N),*dcl).*

For rules with gaps, it is a little more complex. The contribution of a GG rule with reduction to the derivational context of a symbol

is the pair that contains in its both parts the elements which effectively intervene in the reduction. For instance, the rule :

Rel-marqueur Gap(X) NP ---> Rel-pr Gap(X)

becomes :

*Rel-marqueur(*dc1) Gap(X) NP(*dc2) --->*

*Rel-pr(*dc3) gap(X) Concatenate(*dc2,(Rel-marqueur.NP,Rel-pr),*dc3).*

(Rel-marqueur and NP have the same ascendants.)

The rule :

NP(X) M-gap(Y) prep det gap(Z) prep(of) NP(X) --->

NP(X) prep [whose] gap(Z) M-gap(Y).

remains unchanged because the only elements involved in the reduction are det and prep(of) (which are preterminals) and the other elements are only repositioned.

3.3.2_ Definition of a domain for a given constraint :

For each Ross' constraint Ri, we define a **R-domain** Di, in which any constituent may move without any restriction due to that constraint. The R-domain is the subtree whose root node is the symbol which is the lowest ascendant specified in the constraint. This notion of R-domain is very similar to that of c-domain for LFG. The main difference in our work is that R-domains are not fixed a priori in rules on precise symbols (S, NP, ...) but they may differ from a constraint to another. As we will see later, in GG it is not necessary to mark in advance the symbols that delimit domains. These symbols can be dynamically detected during the parsing process.

If we come back to the examples in section 3.1, the R-domain Di in which A (element of Di) can move without any restriction for the complex NP constraint is the R-domain dominated by the node S. In other words, the root node Ni of Di is S. If we represent the complex NP constraint by the list of the ascendant rules that will prevent A from moving out of Di, if A has these rules in its derivational context, then, we have, for this precise constraint :

NP ---> NP(subj) S
S ---> ... A ...
(... is a kind of gap, it means any sequence of symbols)
This list is the blocking rules of the constraint.
The ... can be viewed as super-gaps ranging over symbols and gaps.

Then, the root node Ni of the R-domain Di for this constraint is the symbol which is in the left hand part of the last rule of this ordered list (here : S ---> ... A ...).

For the sentential subject constraint, the blocking rules that will prevent A from moving out of the R-domain of this constraint are :

S ---> ... NP ...
NP ---> ... S ...
S ---> ... A ...

In this case, the last rule is : S ---> ... A ... and S is the root node Nj of the R-domain Dj linked to this constraint.

Next, we say that a constraint Ri is **R-consistent** for a given GG rule if the right hand part of the rule of the list of blocking rules of Ri unifies with (a part of) the left hand part of the GG rule we consider. For example, if the last blocking rule of Rj is :

VP ---> V ... A ...

then, for the GG rule :

V gap(X) A NP ---> A V gap(X) NP

Rj is consistent because V ... A ... unifies with V gap(X) A NP.

In fact, the blocking rules of Ri may be understood as :
- The right hand part of the last rule of the blocking rules describes a part Pi (or, more precisely, an horizontal slice Pi) of a syntactic tree, or, from a GG rule point of view, a kind of prototype of a left hand part of a GG rule to which the constraint may be applied.
- The other rules are the conditions for an element A of Pi not to be allowed to move out of Pi if these conditions are true. From a GG rule point of view, this means that the application of the GG rule we consider is not allowed if A is designed to move out of Pi. The set of GG rules, R-consistent for a constraint Ri, is independent of the sentences to parse.

So, to summarize , a constraint Ri defines an R-domain and an ordered list of blocking rules. Then, in a given GG rule, R-consistent for Ri, we say that A is not allowed to move out of the R-domain Di defined by a constraint Ri if it has in its derivational context the blocking rules of Ri.

Finally, we say that a constituent X belongs to a R-domain Di with a root node Ni iff Ni is the left hand part of a rule in the derivational context of X. Notice that this does not mean that both X and Ni have exactly the same ascendants up in the tree : X may have come in Di via another GG rules. Similarly, two constituents X and Y belong to the same R-domain Di if they have both Ni as the left hand part of the same rule in their respective derivational contexts.

3.3.3 Detection that a constituent moves or is in relation with constituents out of an R-domain :

In section 1.2, we have stated that in a GG rule only the explicit elements that play a role in the movement or relation described by this rule have to appear in the rule. Thus, we have the following definition :

Let's consider the left hand part of a GG rule which is R-consistent for the constraint Ri. Then a constituent X that belongs to the R-domain Di of the constraint Ri **is in relation with constituents out of Di** (or has a movement linked to constituents out of Di) iff there exists a constituent Y in the left hand part of this rule that does not belong to Di.

This definition is a generalization of the different cases met in Ross' constraints with a non-transformational point of view. We have added in this definition the case where a constituent moves in the R-domain, but whose movement is due to the presence of constituents out of that domain. To clarify the understanding of this definition, notice that the left hand part of a GG rule gives the "natural " or "standard " position of constituents whereas the right hand side explicits the sentence structure after movements or deletions of constituents.

In the definition above, by constituent we mean the explicit constituents and the subset of gaps we call M-gaps (cf. section 1.2). An explicit distinction made between gaps and M-gaps is very usefull here because, given a GG rule, an automatic distinction between gaps and M-gaps is a priori ambiguous. The derivational context of a M-gap is the list of the derivational contexts of the symbols it represents. Gaps have no derivational context.

We say that a constraint Ri is **R-valid** for a given GG rule if Ri is R-consistent for this rule and if there exists a constituent X of Di, R-domain of Ri, which is in relation with elements out of Di. The R-validity of a constraint, for a given GG rule, depends on the sentence being parsed since we need to have the derivational context of each symbol to decide if Ri is R-valid.

3.3.4 Expression of Ross' constraints in a GG rule :

Let's consider a given constraint Ri, R-valid for a given GG rule. All the elements ei of that rule that belong to the R-domain

Di of Ri are a priori concerned by the constraint. Then, for any ei, element of Di, the GG rule is applicable iff the derivational context of ei (dci) does not contain the blocking rules of Ri (bci) . For that purpose, we use the predicate :

NOT-CONTAIN(*dci,*bci).

Depending on the kind of element ei (explicit or M-gap), we have two slightly different treatments :

(1) ei is an explicit constituent : the unification between dci and bci is direct. If all the elements of bci unify with elements of dci, then the GG rule considered is blocked by Ri.

(2) ei is an M-gap. Then, we have two cases :

(2a) The constraint Ri is about any constituent X, then all the elements of the list lei the explicit element derivational contexts the M-gap represents are treated as explained above in (1).

(2b) The constraint Ri is about a precise constituent C (NP, N, ... : as in the A-over-A principle), then, we proceed as in (2a) but even if the blocking rules of Ri are contained in an element ej of the list lei, the GG rule is applied but a new constraint is produced that states that ej has to be different of that precise constituent C. Later in the derivation, if it is not the case, the GG rule will be blocked and as soon as the nature of ej is known.

3.3.5_ Properties and implementation of Ross' constraints :

To express Ross' constraints in GG rules, we need to add to the original formalism :

- a new argument to each symbol, so as to memorize the derivational context of that symbol,
- one or several appropriate constraints at the end of each rule, via the predicate NOT-CONTAIN(-,-), which is directly implementable in PROLOG.

The constraints may be added once for all to each GG rule (static point of view) or dynamically, depending on the sentence being parsed (dynamic point of view). In this latter case, Ross' constraints are viewed as integrity constraints (this idea was suggested to us by Jack Minker) that control the derivations of GG rules.

As we have seen in the previous sections, the R-consistency of a constraint is independent of the sentence being parsed. Thus, it is possible to add to each GG rule the set of R-consistent constraints for that rule. In addition, this adjunction can be done automatically by a mapping mechanism once for all [Sebillot, forthcoming]. For example, the adjunction of the complex NP constraint to the GG rule :

NP(subj) V gap(X) NP(obj) --->
NP(obj) [which] NP(subj) V gap(X)

that describes extrapositions such as [Ross 74] :
"The hat which I believe that Otto was wearing. "

has to block sentences such as :
** "The hat which I believe the claim that Otto was wearing. "*

Thus, the complex NP constraint added to that GG rule produces the following GG rule with constraints :

*NP(subj,*a1) V(partitive,*a2) gap(X) NP(obj,*a3) --->*
*NP(obj,*a3) [which] NP(subj,*a1) V(partitive,*a2) gap(X)*
*NOT-CONTAIN(*a3,(NP,NP(subj).S)).*

*where each *ai denote the derivational context of the symbol it belongs to.*

On the contrary, we have noted that the R-validity of a constraint for a given GG rule depends on the sentence being parsed. From the dynamic point of view, the constraints, viewed as integrity constraints, are taken into account during the parsing process only if they are R-valid. This point of view seems to be more efficient, although no complete implementation of it has been carried out and tested.

The formalism we have presented to express long distance dependency relation constraints is a priori completely independent of the parsing strategy. A top-down version of GG rules including Ross' constraints is being implemented in our laboratory [Sebillot, forthcoming]. A parallel implementation is under study at the University of Maryland in Jack Minker's group on a ZMOB parallel machine [Kasif and al. 83]. This machine accepts a programming language, PRISM, which is very similar to PROLOG. The constraints are put in a special component that interact with the other components that contains the GG rules (IDBs) and the dictionary (EDBs).

Another advantage of our formalism is that we make a clear distinction between the description of a sub-language (designed, for instance, for a man-machine interface) and the description of generalizations about that language. Then, generalizations are defined once for all and automatically added to the sublanguage described. Finally, due to the PROLOG specificities, constraints are expressed via the use of logical variables on which unification is applied. This makes GG rules much more reliable and easy to write.

4 CONCLUSION.

In this paper, we have described how it is possible to add restrictions to gapping grammars in order to have a relevant set of

rules that parses only well formed sentences, even if well-formedness for natural language sentences remains to be defined. We have presented two main classes of restrictions that :

- limit the freedom of expansion of gaps via the adjunction of case arguments to gaps.
- limit the freedom of expansion of establishing relations between non-contiguous constituents in a sentence. We have then formalized a quite large set of constraints : Ross' constraints.

The formalism described is independent of the parsing strategy and only makes use of the fundamental properties and facilities of PROLOG. This work remains still under study and we are concious that a lot of work remains to be done in order to have a complete, relevant and efficient formalism for Gapping Grammars that will be easy to use to describe languages and subsets of languages for natural language front ends.

REFERENCES

[Dahl and Abramson 84] H. ABRAMSON, V. DAHL On Gapping Grammars. Proc of the 2nd logic programming conf. Uppsala Univ.

[Bear 82] J. BEAR Gaps as Syntactic Features. Research report of the Indiana university linguistic club.

[Colmerauer 78] A. COLMERAUER Metamorphosis Grammars. In Bolc Edt Natural language communication with computers. Springer Verlag.

[Dahl 77] V. DAHL Interrogation d'une banque de donnees en Espagnol. These d'universite. Marseille-Luminy GIA.

[Dahl 84] V. DAHL More on Gapping Grammars. Conf. on Fifth Generation Computer systems 1984, Tokyo.

[Kaplan 82] KAPLAN R. M. LFG : A formal system for grammatical representation. In The mental representation of grammatical relations, J. Bresnan Edt. MIT Press.

[Gazdar 82] G. GAZDAR Phrase Structure Grammars, in "The nature of syntactic representation " Jacobson and Pullum Edts, D. Reidel Pub.Co.

[Kasif and al. 83] S. KASIF, M. KOHLI, J. MINKER PRISM: A parallel Inference System for Problem Solving. TR 1243, Dept of computer science, Univ. of Maryland.

[Palmer and al. 83] T.W. FININ, M. PALMER Parsing with logical Variables. Conf. on applied Natural language processing.

[Pereira 80] F. PEREIRA, D. WARREN Definite clause grammars for natural language analysis. A survey of the formalism and a comparison with ATN. Artificial intelligence no 13.

[Pereira 83] F. PEREIRA Logic for natural language analysis. SRI international technical note no 275.

[Popowich 85] POPOWICH F. Unrestricted gapping grammars for ID/LP grammars. Conf. on theoretical aspects of natural language understanding, Halifax N.S.

[Popowich2 85] POPOWICH F. Unrestricted gapping grammars. Proceedings of IJCAI-85, Los Angeles.

[Popowich3 85] POPOWICH F. Unrestricted Gapping Grammars: Theory, implementations and applications. PhD. Thesis, forthcoming.Simon Fraser University, Burnaby, BC Canada

[Ross 74] J.R. ROSS Excerpts from Constraints on variables in

syntax. In "On N. Chomsky, critical essays " Anchor Press NY.
[Saint-Dizier 85] SAINT-DIZIER P. A grammar for handling adjective phrases. In proceedings of the conf. on theoretical aspects of natural language understanding, Halifax N.S. Canada.

Imprimé en France

par

l'Institut National de Recherche en Informatique et en Automatique